Annotation sémantique par classification

Yannick Toussaint, Sylvain Tenier

Annotation sémantique par classification

Yannick Toussaint[1], Sylvain Tenier[1,2]

[1] Laboratoire Lorrain de Recherche en Informatique et ses Applications
BP 239, 54506 Vandoeuvre lès Nancy Cedex, France Prénom.Nom@loria.fr et
http://www.loria.fr/equipes/orpailleur

[2] Institut de l'Information Scientifique et Technique
54514 Vandoeuvre-lès-Nancy, France http://www.inist.fr/uri

Résumé :

Les systèmes actuels d'annotation sémantique exploitent peu les connaissances du domaine et fonctionnent essentiellement du texte vers l'ontologie. Pourtant, il est fréquent qu'un élément dans une page doive être annoté par un concept parce que certains autres éléments de cette même page sont annotés par d'autres concepts. Cet article propose une méthode d'annotation prenant en compte cette dépendance entre concepts, exprimée dans une ontologie sous forme de concepts définis. L'utilisation des logiques de descriptions comme mode de représentation unifié de la structure des documents, de l'ontologie et de l'annotation sémantique du document permet de définir le processus d'annotation comme un mécanisme de classification et d'introduire la notion de classe d'annotation.

Mots-clés : Annotation sémantique, Classification symbolique, Représentation des connaissances, Classes d'annotations, Logiques de Descriptions

1 L'importance des connaissances dans le processus d'annotation sémantique

Pour répondre à l'augmentation constante du nombre de pages web, les moteurs de recherche devraient être capables de fournir des réponses plus précises et gérer des requêtes plus complexes, intégrant les connaissances de l'utilisateur. En microbiologie, par exemple, l'interrogation de bases documentaires sur les interactions entre deux gênes produit des milliers de documents. Pour pouvoir se focaliser sur un type particulier d'interaction, des connaissances caractérisant les interactions connues sont nécessaires. Dans le cadre des pages web, la nécessité d'une description formelle du contenu a conduit à la proposition du Web Sémantique par Tim [1]. Il s'agit d'une extension du web actuel qui doit permettre à des agents logiciels de raisonner sur le contenu des pages.

La formalisation de pages web existantes se définit comme la tâche d'annotation. Une annotation consiste en des métadonnées sur une page ou son contenu. On parle d'*annotation sémantique* lorsque les métadonnées sont des instances de concepts et

de relations entre les concepts qui sont définis précisément, décrits formellement et structurés dans une ontologie. Ce processus d'annotation doit rester dynamique : une annotation statique générée lors de la création ne pourrait pas être modifiée lorsque les connaissances du domaine évoluent. De plus, un utilisateur doit être capable d'annoter des pages par rapport à ses propres connaissances, pas celles de l'auteur ni d'un autre utilisateur. Enfin, étant donné le nombre important de pages web, il est essentiel que l'annotation sémantique soit aussi automatique que possible.

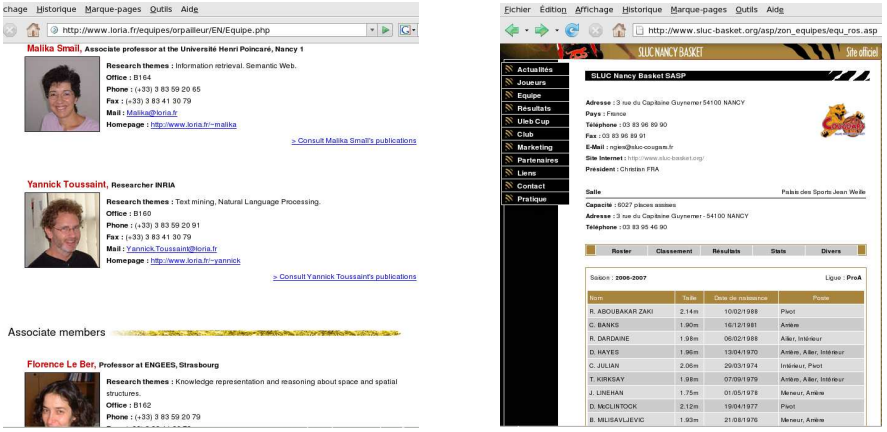

FIG. 1 – Présentation d'équipe de recherche (à gauche) et de basket (à droite)

Comment distinguer les pages de chercheurs d'autres pages web ?

Cette question résume l'idée maîtresse développée dans cet article et elle est illustrée par la Figure 1. En vue de faire de la veille scientifique, l'objectif est d'annoter des pages de chercheurs ou d'équipes de recherche. Des personnes, des publications et des thèmes de recherche sont autant d'indices permettant d'identifier une page comme une page décrivant une équipe de recherche. C'est donc la présence conjointe de ces éléments, définis dans une ontologie, qui permettra au processus d'annotation d'identifier les pages d'équipes de recherche. C'est aussi l'originalité de notre approche comparé aux travaux actuels sur l'annotation sémantique.

Les travaux actuels en annotation sémantique ont pour but d'identifier dans les pages et de formaliser les unités de connaissance correspondant à des concepts et rôles d'une ontologie de domaine. Ces méthodes ne fonctionnent que dans un sens, du texte vers le niveau conceptuel. Elles sont fondées sur des techniques d'extraction d'information (EI) et activent des règles (expressions régulières ou règles plus complexes sur les mots) pour associer des chaînes de caractères de la page web à des concepts de l'ontologie. Des méthodes plus avancées fondées sur l'induction de wrappers [10] exploitent des automates d'arbre générés par apprentissage, tel que [2]. Ces méthodes revêtent un intérêt particulier dans le cadre des documents semi-structurés comme les pages web, dans lesquels la structure est utilisée pour repérer les relations entre instances de concepts. Cela permet ainsi d'associer à la bonne personne, ses thèmes de recherche et ses publications.

Ces travaux négligent cependant les connaissances définies dans l'ontologie. Dans le domaine de la recherche, ces connaissances permettent de déduire de la page d'exemple (Fig. 1) que les personnes sont des chercheurs et que ce regroupement de chercheurs constitue une équipe. Il existe donc une analogie forte entre les éléments présents dans une page web, la manière dont ils sont regroupés, et la définition qui en est donnée dans l'ontologie. Ces concepts sont définis par un ensemble nécessaire et suffisant de conditions. Par exemple, un chercheur est une personne qui a publié des papiers. Tant qu'une personne, 'John', n'a pas publié, elle n'est pas considérée comme un chercheur. Ceci a un impact en retour sur l'annotation d'autres concepts de la page. Si une équipe de recherche est définie comme étant composée de chercheurs, alors le concept `Equipe` sera satisfait ou non pour une page en fonction du concept associé à la chaîne 'John'.

Le problème posé est de générer automatiquement l'annotation d'une page web par des concepts définis. Le processus d'annotation prend en compte l'interaction naturelle illustrée par la Figure 1 qui existe entre la structure de la page web et la sémantique du domaine. Pour cela, nous définissons des modèles de structure représentatifs des structures trouvées dans les pages d'équipes. Une interprétation sémantique est alors "calquée" sur le modèle de structure définissant ainsi la notion de classe d'annotation. Une nouvelle page est ainsi interprétée comme instance de cette classe si elle a la même structure et la même sémantique. Le processus d'annotation ainsi défini devient alors un problème de classification, au sens des logiques de descriptions.

La prochaine section présente l'importance de la structure dans l'interprétation d'une page web. Les opérations de classification permises par la formalisation de cette structure sont introduites en section 3. La section 4 décrit la formalisation du domaine dans une ontologie. La section 5 présente une application de classification par l'introduction de classes d'annotation. Un état de l'art sur les techniques d'identification des concepts primitifs est présenté en section 6.

2 Rôle de la structure dans l'interprétation d'une page

Le cadre applicatif de notre travail est la veille scientifique. En conséquence, nous annotons des pages d'équipes de recherche pour formaliser les connaissances qu'elles contiennent. La Figure 2 présente une visualisation de l'annotation voulue pour une telle page. L'ensemble des éléments constitutifs d'un chercheur sont présents : personnes, thèmes et publications. Le problème est d'établir les relations entre ces éléments. Classiquement, dans un texte, l'analyse de la langue permet de déterminer ces relations. Par exemple, l'analyse de la phrase "Jean a publié un article sur la représentation des connaissances" illustre que Jean est un chercheur. Dans le cadre des pages web, les pages d'équipes contiennent relativement peu de langage naturel. Au contraire, le contenu est structuré en blocs, listes et autres tables. Cette structure permet de rapprocher des éléments identifiés : chaque personne est associée à son mail et à son thème (et non celui d'une autre personne) par appartenance à une même structure. Enfin, l'association de ces éléments permet d'identifier que la personne est en fait un chercheur.

Une caractéristique importante des pages d'équipes est que **la structuration est faite de manière régulière**. Il est ainsi possible de définir un modèle de structure et de l'associer à un contenu. Repérer un chercheur dans une page pour laquelle un modèle existe

```
<body>
<h1>Sem-an-tik</h1>
<div><h2>Jean</h2><p>
<em>Représentation</e
m><a>Publications</a>
</p></div>
<div><h2>Jules</h2><p
><em>Web Sémantique</
em><a>Publications</a
></p></div>
</body>
```

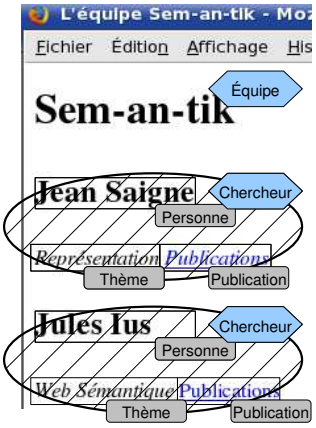

FIG. 2 – Page d'équipe annotée

revient donc à identifier la structure modélisée[1]. Ce modèle peut par exemple décrire un tableau, dans lequel chaque ligne correspond à un chercheur car l'ensemble des éléments définitoires sont présents dans la ligne. Dans l'exemple (Fig. 2), le modèle doit permettre d'identifier que les parties de la page entre chaque et présentent un chercheur. Le modèle doit également définir comment chaque élément définitoire du chercheur est modélisé, à son tour, à l'intérieur de la structure.

Cette modélisation est effectuée grâce à une autre caractéristique des pages web : pour toute page web décrite dans le langage HTML il existe une **représentation arborescente**. Cette représentation arborescente permet de manipuler n'importe quelle structure en utilisant des opérations standard sur les arbres. Le fait de déterminer qu'une page correspond à un modèle revient donc à vérifier qu'elle contient des sous-arbres équivalents à l'arbre servant de modèle. Cette structure arborescente est définie par le Document Object Model (DOM). Le DOM est une interface standard de traitement de documents arborescents. Ce modèle définit chaque objet de la page web comme un noeud et explicite comment les noeuds sont reliés entre eux. Dans le cadre de cet article, deux types de noeuds sont utilisés. Les chaînes de caractères sont représentées dans des *noeuds textes*, tandis que chaque balise HTML est représenté par un *noeud élément* typé par le nom de la balise.

Par exemple, considérons l'extrait de la page web dont le code HTML et l'arbre DOM annoté sont présentés Figure 3. Les balises `<div>`, `<h2>`, `<p>`, `<em>` et `<a>` sont transformées en noeuds éléments. `Jules`, `Jules@semantik.fr` et `Representation` sont transformés en noeuds textes. Ces noeuds sont arrangés dans un arbre ordonné dans lequel chaque arc représente une relation parent/fils entre deux noeuds. Par exemple, le noeud issu de `<em>` est le premier fils du noeud issu de `<p>`. Les noeuds de texte sont typés par la chaîne de caractères qu'ils représentent, les noeuds d'éléments par le nom de la balise HTML. Les relations parent/fils entre ces noeuds sont décrites par les

[1] il est cependant nécessaire de valider la sémantique des éléments identifiés

```
<div>
 <h2>Jean</h2>
 <p>
  <i>Représentation</i>
  <a>Publications</a>
 </p>
</div>
```

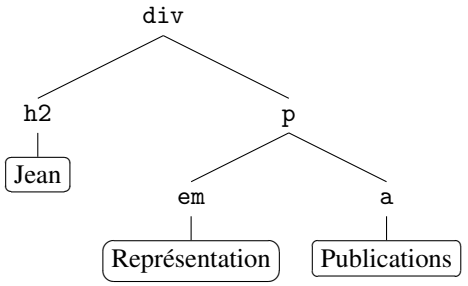

FIG. 3 – Code HTML et représentation arborescente DOM

primitives suivantes :

- `firstChild`, écrit `fC`, identifie le premier fils d'un noeud.
- `nextSibling`, écrit `nS`, identifie le premier frère d'un noeud.

La définition d'un modèle de structure se fait par la description d'un sous-arbre à partir du noeud élément racine du sous-arbre. Toute page HTML peut être décrite comme un arbre binaire à partir de ces deux primitives. Par exemple, la description de l'extrait de la Figure 3 est la suivante : le noeud `div` a pour premier fils `h2` qui a comme premier frère `p` qui a comme premier fils `em` qui a comme premier frère `a`. Pour décrire correctement des modèles de documents, il est nécessaire que des sous-arbres ne puissent se substituer à des noeuds. Il faut pouvoir exprimer qu'un noeud n'a ni fil ni frère. Nous définissons donc les attributs suivants :

- `noChild`, écrit `noC`, identifie un noeud sans fils.
- `noSibling`, écrit `noS`, identifie un noeud sans frère

Dans l'exemple, `a` n'a ni fils ni frère. Un modèle reconnaissant le sous-arbre de l'exemple peut s'écrire ainsi

```
Struct : div and
         fC(h2 and
            nS(p and
               fC(em and
                  nS(a and
                     noC(a) and
                     noS(a)))))
```

Le codage d'un arbre DOM en logiques de descriptions (LD) est immédiat et permet d'accéder aux mécanismes de classification proposés par ce formalisme. La section suivante présente la formalisation des modèles et des pages en LD, ainsi que les opérations de classification nécessaires à la reconnaissance d'une page par un modèle.

3 Classification de pages web par rapport à un modèle décrit dans une logique de descriptions

Un modèle de structure est formalisé en LD avec les concepts et rôles suivants :
- Les noeuds éléments sont représentés sous la forme d'un concept de structure nommé par le nom de la balise. `div`, `h2`, `p`, `em` et `a` sont de tels concepts. Ces concepts sont primitifs.
- Une relation parent/premier_fils entre un concept `A` et un concept `B` est formalisée par le rôle `fC` tel que : $\texttt{A} \sqsubseteq \exists \texttt{fC}.\texttt{B}$
- Une relation a_prochain_frère entre `A` et `B` est formalisée par le rôle `nS` tel que : $\texttt{A} \sqsubseteq \exists \texttt{nS}.\texttt{B}$
- L'absence d'un fils pour un noeud `A` est représentée par un attribut `noC` tel que $\texttt{A} \sqsubseteq \exists \texttt{noC}$ et l'absence d'un frère par un attribut `noS`, tel que $\texttt{A} \sqsubseteq \exists \texttt{noS}$.

Cette formalisation permet de définir une hiérarchie à partir de la relation d'ordre partiel entre les modèles. Prenons les modèles suivants en exemple :

1. $\texttt{Mod1} \equiv \texttt{body} \sqcap \exists \texttt{fC}.\texttt{h1}$
2. $\texttt{Mod2} \equiv \texttt{body} \sqcap \exists \texttt{fC}.(\texttt{h1} \sqcap \exists \texttt{noC} \sqcap \exists \texttt{noS})$
3. $\texttt{Mod3} \equiv \texttt{body} \sqcap \exists \texttt{fC}.(\texttt{h1} \sqcap \texttt{nS}.\texttt{div})$

`Mod1` décrit un modèle sous la forme d'un concept qui prend comme instance toute page web commençant par un titre de niveau 1 (en HTML : la balise $<$ `body` $>$ est suivie de la balise $<$ `h1` $>$). `Mod2` a comme instances les pages qui ne contiennent que ce titre, sans autre balise. Enfin, `Mod3` instancie les pages commençant par un titre de niveau 1 suivi d'un élément de bloc .

`Mod1` est donc plus général que `Mod2` et `Mod3`, tandis que `Mod2` et `Mod3` sont disjoints : il n'est pas possible pour une page d'être instance des concepts `Mod2` et `Mod3`. Par contre, si une page est instance de `Mod2` ou de `Mod3`, elle est forcément instance de `Mod1`. On a donc : $\texttt{Mod2} \sqsubseteq \texttt{Mod1}$, $\texttt{Mod3} \sqsubseteq \texttt{Mod1}$ et $\texttt{Mod2} \sqcap \texttt{Mod3} \sqsubseteq \bot$.

Pour classifier une page web par rapport aux modèles, les noeuds éléments du DOM de la page sont formalisés comme individus de la LD. Pour chaque noeud élément `x`, un individu $\texttt{i}_\texttt{x}$ est généré. Si le noeud a un frère (respectivement un fils) `y`, le rôle $\texttt{nS}(\texttt{i}_\texttt{x}, \texttt{i}_\texttt{y})$ (resp. $\texttt{fC}(\texttt{i}_\texttt{x}, \texttt{i}_\texttt{y})$) est instancié avec son noeud frère $\texttt{i}_\texttt{y}$. S'il n'a pas de frère, l'attribut `noS` (resp.`noC`)) est associé à $\texttt{i}_\texttt{x}$. La formalisation de l'arbre DOM de la Figure 3 est présentée[2] dans la Figure 4.

L'opération permettant la reconnaissance d'une structure de page par un modèle est la classification des individus. Une page est reconnue par un modèle si un individu formalisé depuis son DOM est instance du concept définissant le modèle. Ainsi, l'exemple de la Figure 4 instancie les modèles `Mod1` et `Mod3` mais pas `Mod2`. En effet, `Mod2` définit que la balise <h1> n'a ni fils ni frère ($\texttt{h1} \sqcap \exists \texttt{noC} \sqcap \exists \texttt{noS}$)). Or, dans la page de l'exemple, <h1> a un frère ($\texttt{h1}(\texttt{i}_2) \sqcap \texttt{nS}(\texttt{i}_2, \texttt{i}_3)$).

Pour permettre une annotation correcte, un modèle doit être instancié par l'ensemble des pages à annoter mais être suffisamment spécifique pour ne pas l'être par des pages qui ne doivent pas être annotées. En effet, une page est annotée en fonction du modèle le

[2] $\texttt{i}_8$ représente le noeud racine du deuxième chercheur, non détaillé dans un souci de concision

$\texttt{body}(\texttt{i}_1) \sqcap \texttt{fC}(\texttt{i}_1, \texttt{i}_2)$
$\texttt{h1}(\texttt{i}_2) \sqcap \texttt{nS}(\texttt{i}_2, \texttt{i}_3)$
$\texttt{div}(\texttt{i}_3) \sqcap \texttt{fC}(\texttt{i}_3, \texttt{i}_4) \sqcap \texttt{nS}(\texttt{i}_3, \texttt{i}_8)$
$\texttt{h2}(\texttt{i}_4) \sqcap \texttt{nS}(\texttt{i}_4\texttt{i}_5) \sqcap \texttt{noC}(\texttt{i}_4)$
$\texttt{p}(\texttt{i}_5) \sqcap \texttt{fC}(\texttt{i}_5, \texttt{i}_6) \sqcap \texttt{noS}(\texttt{i}_5)$
$\texttt{em}(\texttt{i}_6) \sqcap \texttt{nS}(\texttt{i}_6, \texttt{i}_7) \sqcap \texttt{noC}(\texttt{i}_6)$
$\texttt{a}(\texttt{i}_7) \sqcap \texttt{noC}(\texttt{i}_7) \sqcap \texttt{noS}(\texttt{i}_7)$
$\texttt{div}(\texttt{i}_8) \sqcap \texttt{fC}(\texttt{i}_8, \texttt{i}_9) \sqcap \texttt{noS}(\texttt{i}_8)$
[...]

FIG. 4 – Arbre DOM de la Figure 3 formalisé en individus d'une LD

plus spécifique qu'elle instancie dans la hiérarchie des modèles. Par exemple, le modèle $\texttt{Mod4} \equiv \texttt{body}$ est instancié par toutes les pages qui ont un contenu, donc toutes les pages d'équipes, mais aussi toutes les autres pages HTML bien formées ! Un modèle correct est le modèle le plus spécifique possible décrivant les balises qui sont obligatoirement présentes (à la bonne position) dans la page et celles qui sont interdites pour qu'une page soit reconnue. Un tel modèle pourrait être, pour la page de la Figure 3 : $\texttt{Mod5} \equiv \texttt{div} \sqcap \exists\texttt{fC}(\texttt{h2} \sqcap \exists\texttt{nS}(\texttt{p} \sqcap \exists\texttt{fC}(\texttt{em} \sqcap \exists\texttt{nS}(\texttt{a} \sqcap \exists\texttt{noC}(\texttt{a}) \sqcap \exists\texttt{noS}(\texttt{a})))))$.

Ce modèle limite l'instanciation de pages non correctes : une page qui aurait exactement la même structure que la page d'exemple mais dont la balise <a>, représentée par le concept a, aurait un frère ne serait pas instance de Mod5. Nous pouvons noter que $\texttt{i}_2$ et $\texttt{i}_8$ sont tous deux instances de Mod5. Cela est du au fait que la définition du concept ne pose aucune contrainte sur le fait que la balise ait un frère ou non. La représentation en logiques de descriptions permet donc de définir précisément des modèles en fonction d'une structure d'arbre et ainsi d'annoter les pages par une simple opération de classification. Cette formalisation s'avère pertinente pour la représentation de structures régulières. Les sections suivantes introduisent la formalisation du domaine de la recherche permettant d'opérer cette classification à la fois sur la structure d'une page et sur la sémantique de son contenu.

4 Ontologie de la recherche

Une ontologie de domaine est définie avec les concepts et rôles suivants :

1. Des concepts qui sont ordonnés par la relation de subsomption dans une hiérarchie H. Chaque concept est nommé par une chaîne de caractères. Un sous-ensemble de H est l'ensemble des concepts *primitifs*. Un concept primitif est défini par une relation de subsomption $\sqsubseteq$ avec un autre concept primitif ou $\top$.
2. Des rôles qui définissent une relation binaire entre deux concepts. Dans l'ontologie $\mathcal{O}$, un seul rôle peut exister entre deux même concepts. Soient $\texttt{C}, \texttt{D} \in H$,

l'existence d'une relation entre C et D est formalisé comme suit : $\texttt{C} \sqsubseteq \exists r.D$.

3. Des concepts *définis* qui sont définis par la relation d'équivalence $\equiv$. L'ensemble des rôles associés à un concept défini C agit comme l'ensemble des conditions nécessaires et suffisantes pour qu'un individu soit une instance de C.

Nous présentons un extrait de l'ontologie $\mathcal{O}$ du domaine de la recherche en fonction de laquelle les pages d'équipes sont annotées :

Concepts primitifs : Personne, Theme, Publication

Concepts définis : Chercheur, Equipe
- $\texttt{Chercheur} \equiv \texttt{Personne} \sqcap \exists r_1.\texttt{Theme} \sqcap \exists r_2.\texttt{Publication}$
- $\texttt{Equipe} \equiv \exists r_3 \texttt{Chercheur}$

5 Application à des classes d'annotation

Une classe d'annotation (CA) formalise l'annotation d'une page web sous la forme d'un concept défini en fonction de l'ontologie de domaine $\mathcal{O}$ et d'un modèle de structure, introduit en section 3 :

1. Soit Mod un concept définissant un modèle composé de n concepts de structure S_1 ... S_n, tels que $Mod \equiv S_1 \sqcap \exists(\texttt{fC} \sqcup \texttt{nS})(S_2))... \sqcap \exists(\texttt{fC} \sqcup \texttt{nS}(S_n))$
2. Soit C un concept défini de $\mathcal{O}$ composé de m rôles, $m < n$ tel que $\texttt{C} \equiv D_0 \sqcap \exists r_1.D_1... \sqcap \exists r_m.D_m$

Alors on définit la classe d'annotation CA définissant l'annotation d'une page instance de Mod par le concept C comme suit :
$\texttt{CA} \equiv S_1 \sqcap \exists \texttt{annotePar.C}$ avec $\forall r_x \in \texttt{C}, \exists(S_y \sqcap \exists \texttt{annotePar.D}_\texttt{x})$

Une CA définit donc que le noeud racine d'un sous-arbre décrit par un modèle est annoté par une instance d'un concept défini de l'ontologie de domaine si ce sous-arbre contient des noeuds annotés par tous les éléments définitoires du concept défini. Pour permettre la modélisation de cette relation entre structure et sémantique, la structure de la page et l'ontologie sont décrites dans un langage commun. Il s'agit du langage OWL-DL fondé sur les logiques de descriptions et pour lequel des raisonneurs ont été développés. Nous avons sélectionné Pellet [14] qui est un raisonneur spécifique pour OWL-DL qui supporte les opérations de classification sur les individus. Le processus d'annotation d'une page par rapport à une CA se fait en trois étapes :

1. Génération des individus à partir du DOM de la page à annoter
2. Classification des individus en ne tenant compte que de la structure
3. Classification sur la sémantique des individus

Nous détaillons ce processus en prenant comme exemple une page à annoter issue de l'exemple Figure 2 et une classe d'annotation CA5 définie à partir du concept $Mod5$, introduit en section 3 et qui intègre la sémantique de l'ontologie $\mathcal{O}$ comme suit :
$\texttt{CA5} \equiv \texttt{div} \sqcap \exists \texttt{annotePar.Chercheur} \sqcap \exists \texttt{fC(h2} \sqcap \exists \texttt{annotePar.Personne}$
$\sqcap \exists \texttt{nS(p} \sqcap \exists \texttt{fC(em} \sqcap \exists \texttt{annotePar.Theme}$
$\sqcap \exists \texttt{nS(a} \sqcap \exists \texttt{annotePar.Publication} \sqcap \exists \texttt{noC(a)} \sqcap \exists \texttt{noS(a)})))$. La Figure 5 présente la visualisation sous forme d'arbre de CA5 et de la page à annoter.

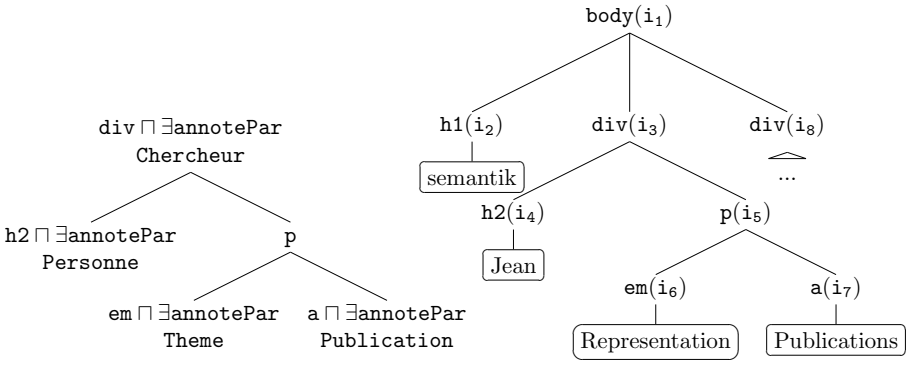

FIG. 5 – Classe d'annotation et page à annoter

Génération des individus

La première étape consiste à transformer en individus de la LD le code HTML de la page à annoter. En termes d'implémentation, cette transformation est effectuée par un script exploitant la librairie ruby *hpricot* qui génère automatiquement un arbre DOM à partir du code HTML d'une page web puis génère les individus à partir de cet arbre. La sortie est un fichier .owl décrivant les individus dans le langage OWL-DL. La formalisation de la page d'exemple sous forme d'individus est présentée Figure 4.

Classification sur la structure

La deuxième étape consiste à classifier les individus par rapport aux concepts de structure définis dans les CA. Ce mécanisme de classification est décrit en section 3. En pratique, le raisonneur Pellet est appliqué. La sortie est l'ensemble des CA dont le modèle de structure est instancié par les individus. Dans l'exemple, les individus sont classifiés comme instances des concepts de structure de `CA5`

Classification sur l'ontologie

Enfin la troisième étape consiste à classifier les individus en fonction la sémantique définie dans les CA par le rôle `annotePar`. Pour chaque individu $\mathtt{i}_x$ classifié comme instance d'un concept de structure pour lequel le rôle $\exists\mathtt{annotePar}.\mathtt{C}$ est défini, l'instanciation du rôle est validée en fonction du concept $\mathtt{C} \in \mathcal{O}$ codomaine de `annotePar`. Dans l'exemple, les individus concernés sont :

1. $\mathtt{i}_3$, instance d'un `div` annoté par le concept défini de `Chercheur`
2. $\mathtt{i}_4$, instance d'un `h2` annoté par le concept primitif de `Personne`
3. $\mathtt{i}_6$, instance d'un `em` annoté par le concept primitif de `Theme`
4. $\mathtt{i}_7$, instance d'un `a` annoté par le concept primitif de `Publication`

Si le concept C est primitif, le noeud texte de l'individu à classifier $\mathtt{i_x}$ est analysé syntaxiquement. Si le texte est reconnu comme instance de C, le rôle annotePar est instancié et ajouté à la définition de $\mathtt{i_x}$. Dans l'exemple, $\mathtt{i_4}$ a pour noeud texte la chaîne de caractères "Jean". Des connaissances sur le domaine permettent de déterminer qu'il s'agit d'un prénom, qui identifie bien une personne. La définition de $\mathtt{i_4}$ est donc étendue comme suit : $\mathtt{div}(\mathtt{i_3}) \sqcap \mathtt{fC}(\mathtt{i_3}, \mathtt{i_4}) \sqcap \mathtt{nS}(\mathtt{i_3}, \mathtt{i_8}) \sqcap \mathtt{annotePar}(\mathtt{i_4}, \mathtt{Personne})$ La même opération est effectuée pour $\mathtt{i_5}$ et $\mathtt{i_6}$. Au final, on a :

$$\mathtt{em}(\mathtt{i_6}) \sqcap \mathtt{nS}(\mathtt{i_6}, \mathtt{i_7}) \sqcap \mathtt{noC}(\mathtt{i_6}) \sqcap \mathtt{annotePar}(\mathtt{i_6}, \mathtt{Theme})$$
$$\mathtt{a}(\mathtt{i_7}) \sqcap \mathtt{noC}(\mathtt{i_7}) \sqcap \mathtt{noS}(\mathtt{i_7}) \sqcap \mathtt{annotePar}(\mathtt{i_7}, \mathtt{Publication})$$

Si le concept C est défini, la définition du concept est obtenu par interrogation de l'ontologie $\mathcal{O}$. Dans l'exemple, $\mathtt{i_3}$ est instance d'un div dont le rôle annotePar doit être instancié par le concept de Chercheur. L'interrogation de $\mathcal{O}$ retourne :
$\mathtt{Chercheur} \equiv \mathtt{Personne} \sqcap \exists r_1.\mathtt{Theme} \sqcap \exists r_2.\mathtt{Publication}$
Personne étant instancié par $\mathtt{i_2}$, Theme par $\mathtt{i_5}$ et Publication par $\mathtt{i_6}$, le concept de chercheur est validé et la définition de $\mathtt{i_3}$ est étendue comme suit :
$\mathtt{div}(\mathtt{i_3}) \sqcap \mathtt{fC}(\mathtt{i_3}, \mathtt{i_4}) \sqcap \mathtt{nS}(\mathtt{i_3}, \mathtt{i_8}) \sqcap \mathtt{annotePar}(\mathtt{i_3}, \mathtt{Chercheur})$
Au final, l'ensemble de la définition de CA5 est classifiée, ce qui permet d'annoter la page en fonction de cette CA.

6 Identification de concepts primitifs

La classification sur l'ontologie nécessite d'identifier des objets textuels dans la page pour instancier les concepts primitifs. Nous utilisons pour cela des techniques existantes en annotation sémantique, issues de travaux en Recherche et Extraction d'Information. Un état de l'art détaillé a été dressé par [15]. Les systèmes actuels peuvent être classifiés en quatre catégories :

1. **Annotation centrée sur l'utilisateur** Ces systèmes interactifs s'appuient sur des interfaces qui affichent simultanément l'ontologie et la page à annoter. L'utilisateur marque alors dans la page les différentes instances de concepts de l'ontologie. En sortie, le système génère un fichier contenant les annotations. Amaya [13] et Mangrove [11] sont des outils spécifiques à l'annotation de pages web qui permettent la population d'ontologie depuis une page web à l'intérieur d'un navigateur web.
2. **Systèmes fondées sur l'apprentissage** Ces systèmes sont entraînés pour une tâche particulière, comme l'extraction du contenu de rapports financiers ou des dépêches d'agence. Dans une première étape, le système apprend des règles d'extraction à partir d'un ensemble de documents annotés. Puis de nouveaux documents sont annotés en appliquant les règles de manière non supervisée. De tels systèmes sont évalués dans des séries d'évaluations, telles que le MUC [3] ou le concours ICDM (Conférence Internationale sur la Fouille de Données)
3. **Systèmes semi-automatiques** Ces systèmes intègrent des méthodes des deux approches précédentes. Ils sont toujours centrés sur l'utilisateur mais automatisent

certaines taches. Ainsi, S-CREAM [8] intègre le module d'Extraction d'Information Amilcare [4] : des annotation crées par l'utilisateur sont utilisées comme entrée d'un algorithme d'apprentissage. Le résultat permet au système de suggérer des annotations lors de l'annotation d'un nouveau document. Lixto [7] et SHOE [9] exploitent des règles qui reconnaissent des chaînes de caractères dans un document à partir d'une liste de termes associés à des concepts de l'ontologie.

4. **Annotation non supervisée** L'objectif des systèmes non supervisés est de fonctionner sans intervention humaine. Ces systèmes parcourent le Web et exploitent la redondance de l'information pour valider les annotations. Cette approche nécessite toutefois une large quantité de données pour fonctionner. Armadillo [12] associe des techniques d'extraction d'information à une méthode statistique d'intégration d'information pour confirmer la validité des connaissances extraites. [5] a étendu la méthode pour permettre la détection des relations. KnowItAll [6] définit une mesure à partir des résultats fournis par différents moteurs de recherche.

Notre implémentation reprend les principes des systèmes semi-automatiques. Après détection des chaînes connues, l'utilisateur complète et corrige les éléments détectés correspondant à des instances de concepts primitifs.

7 Conclusions et perspectives

Nous avons présenté un système d'annotation sémantique fondé sur une analogie entre la représentation des connaissances d'un domaine par des concepts définis dans une ontologie et l'interprétation de la structure d'une page web pour repérer les relations entre les éléments présentés dans la page. La formalisation de cette analogie dans des classes d'annotation permet d'annoter automatiquement les pages en fonction de leur structure et de leur sémantique. Le système a été testé sur quelques pages réelles (les pages présentant les équipes du LORIA et du réseau d'excellence Knowledge Web). Ces tests ont permis de cerner les principales limites du système :

1. L'écriture des classes d'annotation est longue et nécessite une expertise en logiques de description.
2. Les classes d'annotation ne peuvent être écrites que pour les structures dans lesquelles l'ensemble des instances de concepts primitifs sont présents dans un même sous-arbre.

L'approche consistant à définir un processus d'annotation par classification plutôt que comme un mécanisme de règles ouvre des pistes intéressantes. Cela permet d'exploiter des connaissances du domaine et de donner une vision claire, formelle, de ce qu'est une annotation et du processus de génération. Nos travaux en cours visent à aller plus loin dans l'exploitation des logiques de descriptions pour la représentation de la structure des page web. Par exemple, une table et une liste à puce exprimant des connaissances équivalentes devraient être formalisées par un même concept. Enfin, une classe d'annotation devrait pouvoir être générée automatiquement à partir de l'annotation partielle d'une page web, afin d'annoter automatiquement l'ensemble de la page ainsi que les pages classifiées comme instances de la classe.

Références

[1] BERNERS-LEE T. (1999). *Weaving the Web : The Original Design and Ultimate Destiny of the World Wide Web by Its Inventor*. Harper San Francisco.

[2] CARME J., GILLERON R., LEMAY A. & NIEHREN J. (2007). Interactive learning of node selecting tree transducers, machine learning. *Machine Learning*, **66**(1), 33–67.

[3] CHINCHOR N. (1997). Overview of MUC-7. In *Proceedings of the Seventh Message Understanding Contest*. Fairfax, VA, USA.

[4] CIRAVEGNA F., DINGLI A., WILKS Y. & PETRELLI D. (2002). Amilcare : adaptive information extraction for document annotation. In *SIGIR '02 : Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, p. 367–368, New York, NY, USA : ACM Press.

[5] DE BOER V., VAN SOMEREN M. & WIELINGA B. J. (2006). Extracting instances of relations from web documents using redundancy. In *ESWC European Semantic Web Conference*, p. 245–258.

[6] ETZIONI O., CAFARELLA M., DOWNEY D., KOK S., POPESCU A. M., SHAKED T., SODERLAND S., WELD D. S. & YATES A. (2004). Web-scale information extraction in knowitall : (preliminary results). In *WWW '04 : Proceedings of the 13th international conference on World Wide Web*, p. 100–110, New York, NY, USA : ACM Press.

[7] GOTTLOB G., KOCH C., BAUMGARTNER R., HERZOG M. & FLESCA S. (2004). The lixto data extraction project : back and forth between theory and practice. In *PODS '04 : Proceedings of the twenty-third ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, p. 1–12, New York, NY, USA : ACM Press.

[8] HANDSCHUH S., STAAB S. & CIRAVEGNA F. (2002). S-cream-semi-automatic creation of metadata. *Proc. of the European Conference on Knowledge Acquisition and Management*.

[9] HEFLIN J., HENDLER J. A. & LUKE S. (2003). Shoe : A blueprint for the semantic web. In *Spinning the Semantic Web*, p. 29–63.

[10] KUSHMERICK N. (1997). *Wrapper induction for information extraction*. PhD thesis. Chairperson-Daniel S. Weld.

[11] MCDOWELL L., ETZIONI O., GRIBBLE S. D., HALEVY A. Y., LEVY H. M., PENTNEY W., VERMA D. & VLASSEVA S. (2003). Mangrove : Enticing ordinary people onto the semantic web via instant gratification. In *International Semantic Web Conference*, p. 754–770.

[12] NORTON B., CHAPMAN S. & CIRAVEGNA F. (2005). Orchestration of semantic web services for large-scale document annotation. In *ESWC European Semantic Web Conference*, p. 649–663.

[13] QUINT V. & VATTON I. (1997). An introduction to amaya. *World Wide Web J.*, **2**(2), 39–46.

[14] SIRIN E. & PARSIA B. (2004). Pellet : An owl dl reasoner. In V. HAARSLEV & R. MÖLLER, Eds., *Description Logics*, volume 104 of *CEUR Workshop Proceedings* : CEUR-WS.org.

[15] UREN V., CIMIANO P., IRIA J., HANDSCHUH S., VARGAS-VERA M., MOTTA E. & CIRAVEGNA F. (2006). Semantic annotation for knowledge management : Requirements and a survey of the state of the art. *Journal of Web Semantics : Science, Services and Agents on the World Wide Web*, (4), 14–28.